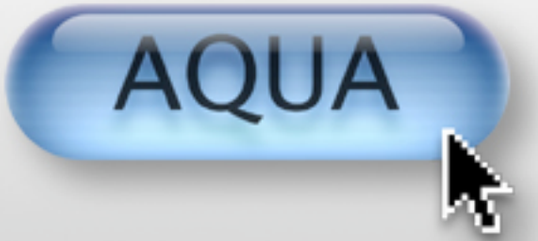
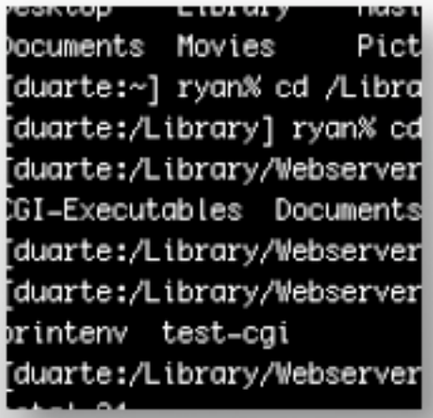
Address Book Framework

Session 417





Address Book Framework

John Geleynse
Manager, Software Evangelism and User Experience Evangelist
geleynse@apple.com

Address Book Framework

Henri Lamiraux
Engineering Manager
lamiraux@apple.com

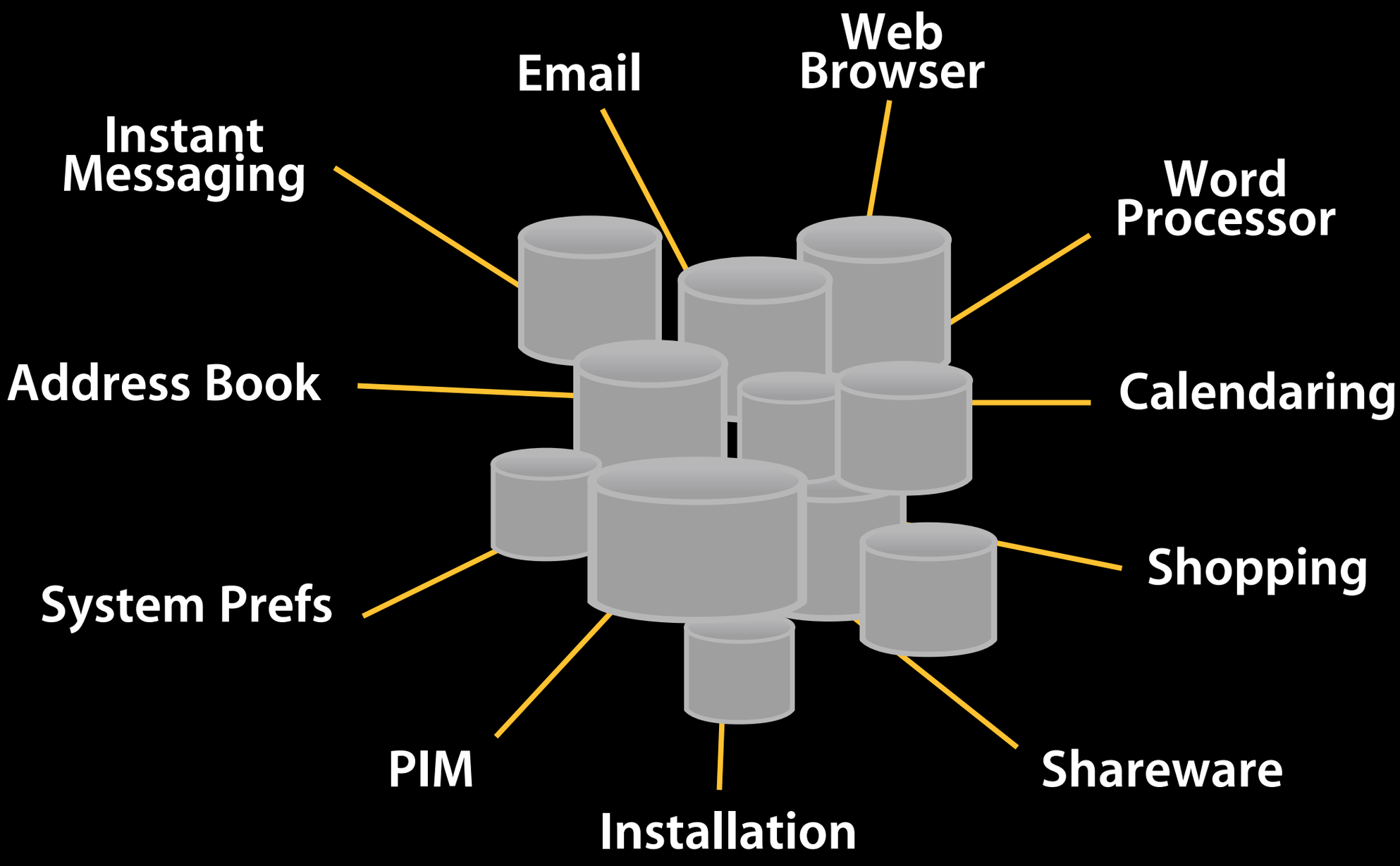
What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - Enhanced AppleScript Support
  - People Picker
  - Action Plug-Ins

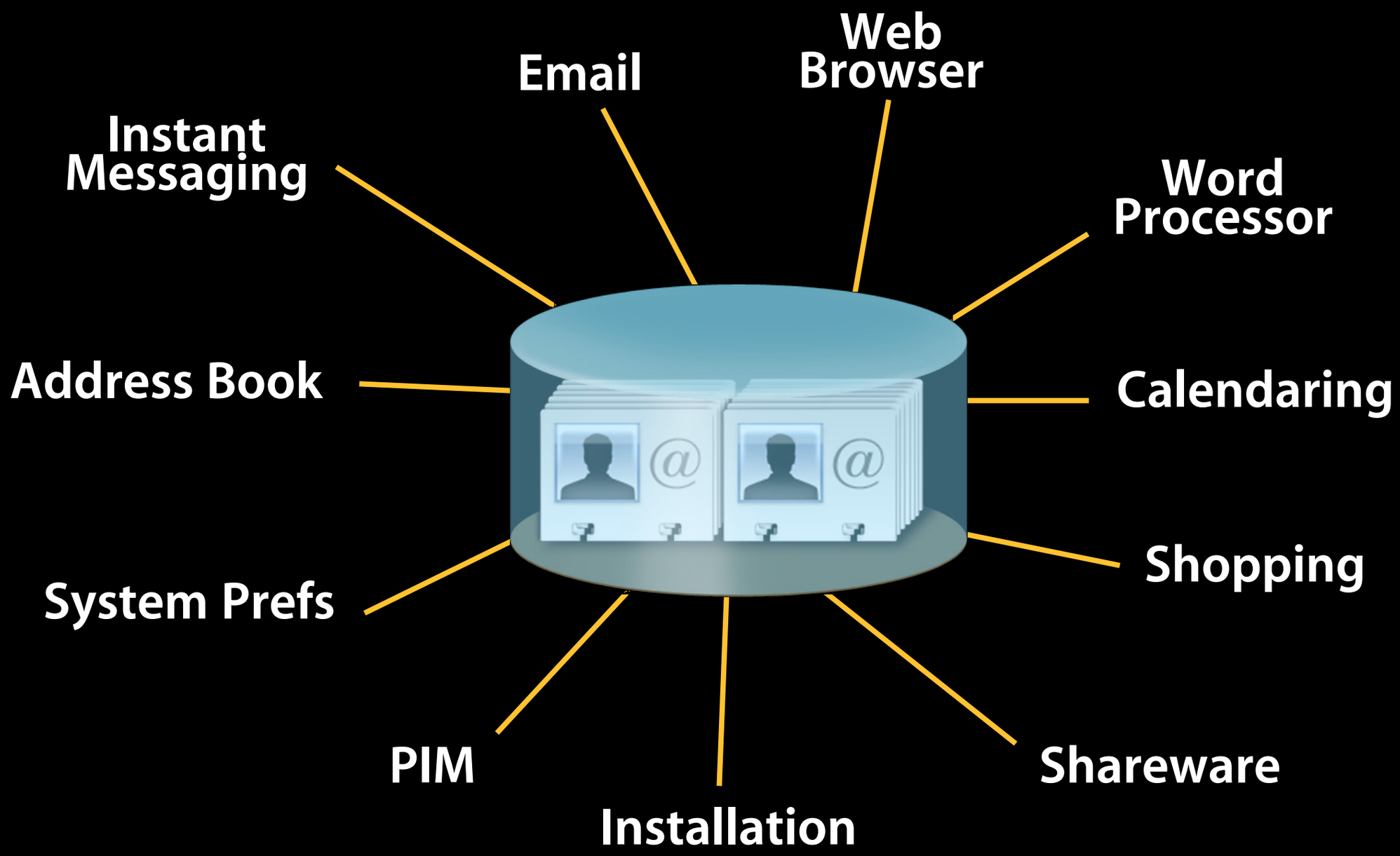
What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - Enhanced AppleScript Support
  - People Picker
  - Action Plug-ins

Before Jaguar



Jaguar—Centralized Storage



Clients

Address Book

Mail

iChat

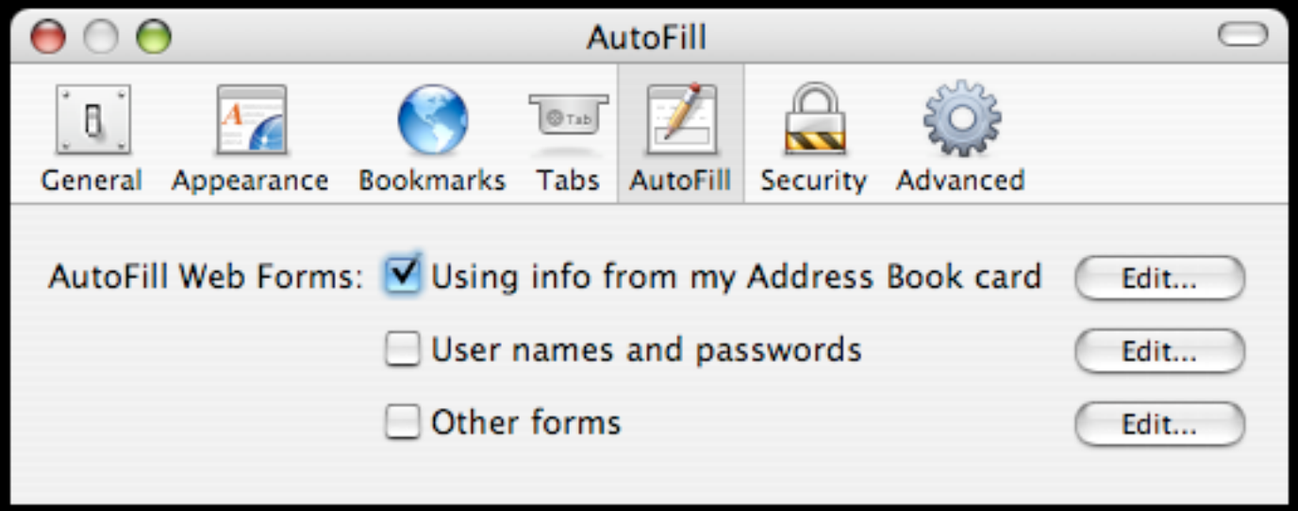
Safari

Sherlock

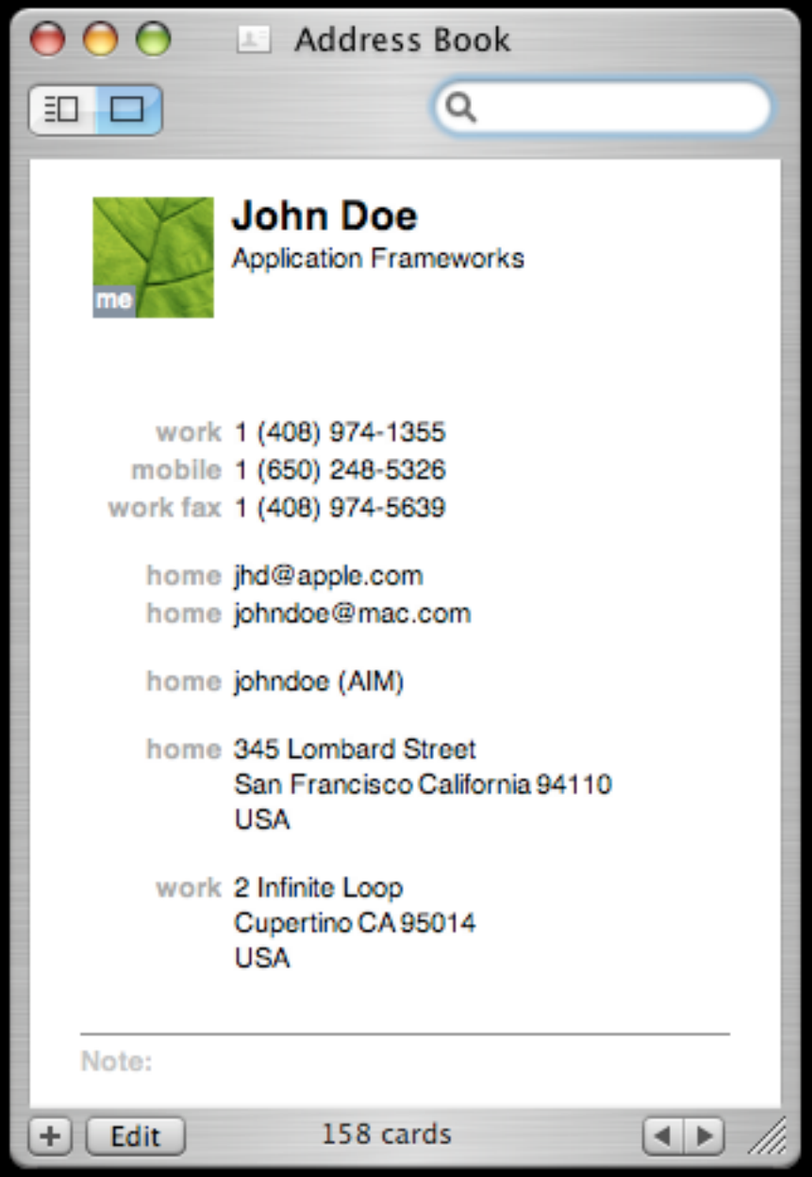
iCal

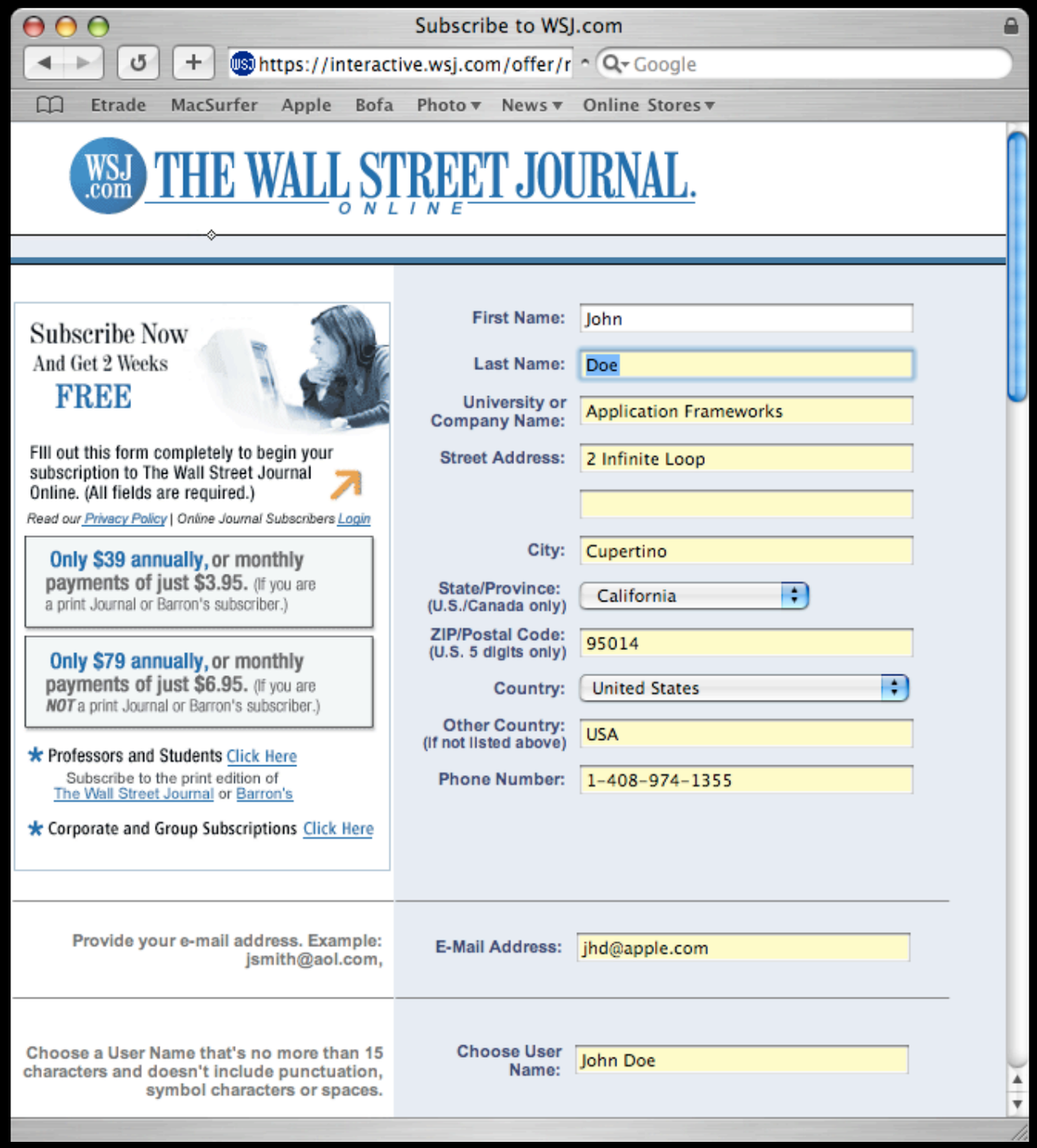
…and many other 3rd party Applications

Safari AutoFill Web Forms



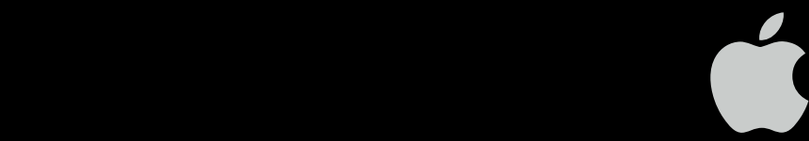
Safari AutoFill Web Forms





Address Book Framework

- Accessible from
  - Objective-C
  - C/C++
  - AppleScript
- Thread Safe
- Concurrent access by multiple applications
- Extensible property set

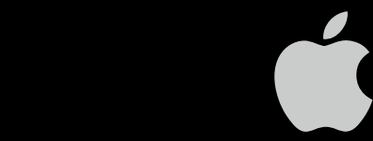


API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

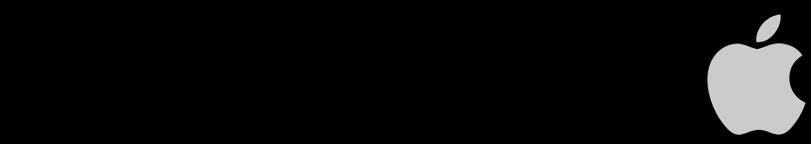
ABRecord

- Row in Address Book Database
- Unique ID
- Abstract base class for:
  - ABPerson
  - ABGroup
- “Dictionary” of Property/Value pairs



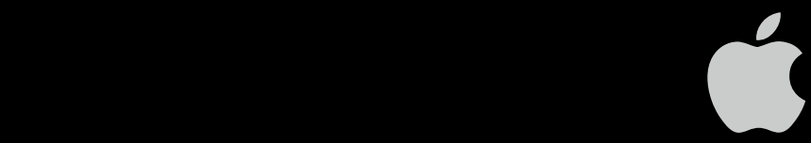
ABRecord

```
-(NSString *)uniqueId;


-(BOOL)setValue:(id)value forProperty:(NSString*)property;
-(id)valueForProperty:(NSString *)property;
-(BOOL)removeValueForProperty:(NSString *)property;
```



ABRecord

```
-(NSString *)uniqueId;


-(BOOL)setValue:(id)value forProperty:(NSString*)property;
-(id)valueForProperty:(NSString *)property;
-(BOOL)removeValueForProperty:(NSString *)property;
```



ABRecord

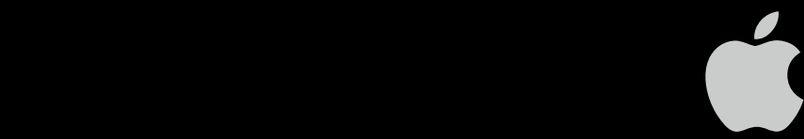
```
ABRecord *record = …;
id value = [record valueForProperty:kSomeProperty];
if (value == nil) {
 <Do something>
} else {
 <Do something else>
}
[record setValue:value forProperty:kSomeProperty];
```

ABRecord

```
ABRecord *record = …;
id value = [record valueForProperty:kSomeProperty];
if (value == nil) {
 <Do something>
} else {
 <Do something else>
}
[record setValue:value forProperty:kSomeProperty];
```

ABRecord

```
ABRecord *record = …;
id value = [record valueForProperty:kSomeProperty];
if (value == nil) {
 <Do something>
} else {
 <Do something else>
}
[record setValue:value forProperty:kSomeProperty];
```



ABRecord

```
ABRecord *record = …;
id value = [record valueForProperty:kSomeProperty];
if (value == nil) {
 <Do something>
} else {
 <Do something else>
}
[record setValue:value forProperty:kSomeProperty];
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

Properties

- Unique Name
  - FirstName, LastName, etc.
- Type
  - Integer, string, etc.

Properties

- Single Value Types

Type	Single Value
string	kABStringProperty
integer	kABIntegerProperty
real	kABRealProperty
date	kABDateProperty
array	kABArrayProperty
dictionary	kABDictionaryProperty
data	kABDataProperty

Properties

- Example of Single Value properties
  - First Name—kABStringProperty
  - Last Name—kABStringProperty
  - Birthday—kABDateProperty

Properties

- Single values don’t work for Emails, Phone Numbers, etc.
- Need to be able to associate multi label/value pairs with one property:
  - Home: jdoe@mac.com
  - Home: jdoe@earthlink.net
  - Work: jdoe@apple.com

Properties

- Introduce notion of multi-value
- ABMultiValue class encapsulates a multi-value

ABMultiValue

- Collection class like NSDictionary and NSArray
- Mutable and non-mutable flavors

ABMultiValue: Array of Triplets

- Label
- Identifier
- Value

ABMultiValue: Array of Triplets

- Label
  - String (e.g., “Home”, “Work”…)
  - Doesn’t have to be unique
    - E.g., multiple “home” email addresses
- Identifier
- Value

ABMultiValue: Array of Triplets

- Label
- Identifier
  - Provides a reference to a specific value within the multi-value
    - Labels are not unique
    - Indexes are not reliable
- Value

ABMultiValue: Array of Triplets

- Label
- Identifier
- Value
  - All values must have the same type
  - E.g., kABMultiStringProperty -> all string
  - E.g., kABMultiDateProperty -> all date

ABMultiValue

Type	Single Value	Multi Value
string	kABStringProperty	kABMultiStringProperty
integer	kABIntegerProperty	kABMultiIntegerProperty
real	kABRealProperty	kABMultiRealProperty
date	kABDateProperty	kABMultiDateProperty
array	kABArrayProperty	kABMultiArrayProperty
dictionary	kABDictionaryProperty	kABMultiDictionaryProperty
data	kABDataProperty	kABMultiDataProperty

Sample Code: ABMultiValue

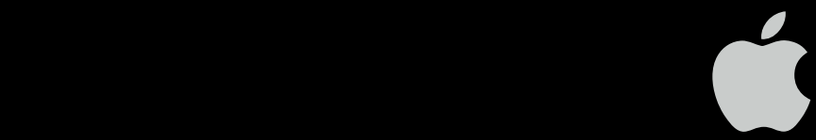
- Display all email addresses of a person

Sample Code: ABMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
                  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
 NSLog(@”label: %@ email: %@”,
          [emails labelAtIndex:i],
          [emails valueAtIndex:i]);
}
```

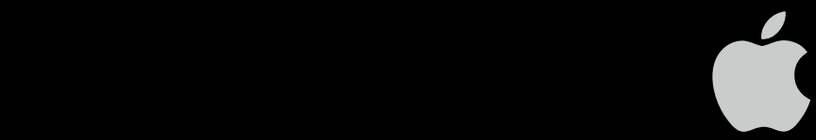
Sample Code: ABMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
                  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
 NSLog(@”label: %@ email: %@”,
          [emails labelAtIndex:i],
          [emails valueAtIndex:i]);
}
```



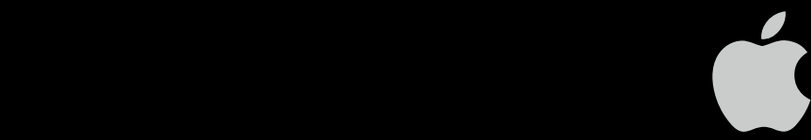
Sample Code: ABMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
                  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
 NSLog(@”label: %@ email: %@”,
          [emails labelAtIndex:i],
          [emails valueAtIndex:i]);
}
```



Sample Code: ABMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
                  valueForProperty:kABEmailProperty];

int count = [emails count];

for (i = 0; i < count; i++) {
 NSLog(@”label: %@ email: %@”,
          [emails labelAtIndex:i],
          [emails valueAtIndex:i]);
}
```



Sample Code: ABMutableMultiValue

- Add an email address to a person

Sample Code: ABMutableMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
valueForProperty:kABEmailProperty];


ABMutableMultiValue *mutableEmails = [emails mutableCopy];


[mutableEmails addValue:@”jdoe@apple.com”” 
                withLabel:kABHomeEmailLabel];


...
```

Sample Code: ABMutableMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
valueForProperty:kABEmailProperty];


ABMutableMultiValue *mutableEmails = [emails mutableCopy];


[mutableEmails addValue:@”jdoe@apple.com”” 
                withLabel:kABHomeEmailLabel];


...
```

Sample Code: ABMutableMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
valueForProperty:kABEmailProperty];


ABMutableMultiValue *mutableEmails = [emails mutableCopy];


[mutableEmails addValue:@”jdoe@apple.com”” 
                withLabel:kABHomeEmailLabel];


...
```

Sample Code: ABMutableMultiValue

```
ABPerson *person = …;

ABMultiValue *emails = [person
valueForProperty:kABEmailProperty];


ABMutableMultiValue *mutableEmails = [emails mutableCopy];


[mutableEmails addValue:@”jdoe@apple.com”” 
                withLabel:kABHomeEmailLabel];


...
```

Sample Code: ABMutableMultiValue

```
...

[person setvalue:mutableEmails forProperty: kABEmailProperty];

[mutableEmails release];
```

Sample Code: ABMutableMultiValue

```
...

[person setvalue:mutableEmails forProperty: kABEmailProperty];

[mutableEmails release];
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

ABPerson

- Subclass of ABRecord
- Represents a person

ABPerson

- Some standard properties

Name	Type
kABFirstNameProperty	kABStringProperty
kABLastNameProperty	kABStringProperty
kABBirthdayProperty	kABDateProperty
kABEmailProperty	kABMultiStringProperty
kABPhoneProperty	kABMultiStringProperty
kABAddressProperty	kABMultiDictionaryProperty
...	...

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

ABGroup

- Subclass of ABRecord
- Represents a group of people or other groups
  - No circular references allowed

ABGroup

- One standard properties

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

ABAddressBook

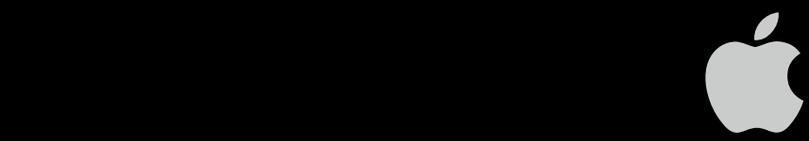
- Represents the Address Book Database
- Located in:

  ~Library/Application Support/AddressBook

- Unique Shared Instance

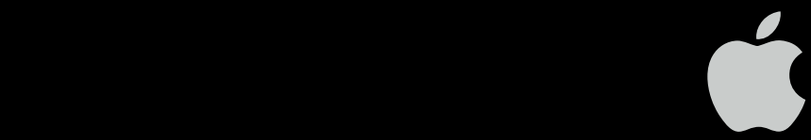
ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;
```



ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;
```



ABAddressBook

```
+ (ABAddressBook *)sharedAddressBook;

- (BOOL)save;
- (BOOL)hasUnsavedChanges;

- (ABPerson *)me;
- (void)setMe:(ABPerson *)moi;
```

ABAddressBook

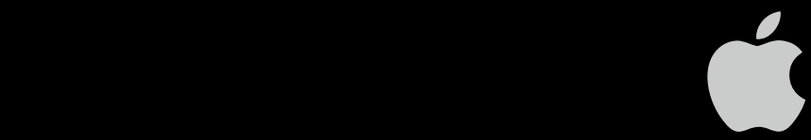
```
- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;

- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

```
- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;

- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```

ABAddressBook

```
- (ABRecord *)recordForUniqueId:(NSString *)uniqueId;

- (BOOL)addRecord:(ABRecord *)record;
- (BOOL)removeRecord:(ABRecord *)record;

- (NSArray *)people;
- (NSArray *)groups;
```



Sample Code: ABAddressBook

- Add a person named “John Doe”

Sample Code: ABAddressBook

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[ABPerson alloc] init];

. . .
```

Sample Code: ABAddressBook

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[ABPerson alloc] init];

. . .
```

Sample Code: ABAddressBook

```
ABAddressBook *addressBook;
ABPerson *newPerson;

addressBook = [ABAddressBook sharedAddressBook];

newPerson = [[ABPerson alloc] init];

...
```

Sample Code: ABAddressBook

```
...
[newPerson setValue:@"John"
  forProperty:kABFirstNameProperty];

[newPerson setValue:@"Doe"
 forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];
[newPerson release];

[addressBook save];
```

Sample Code: ABAddressBook

```
...
[newPerson setValue:@”John”
  forProperty:kABFirstNameProperty];

[newPerson setValue:@”Doe”
 forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];
[newPerson release];

[addressBook save];
```

Sample Code: ABAddressBook

```
...
[newPerson setValue:@"John"
  forProperty:kABFirstNameProperty];

[newPerson setValue:@"Doe"
 forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];
[newPerson release];

[addressBook save];
```

Sample Code: ABAddressBook

```
...
[newPerson setValue:@"John"
  forProperty:kABFirstNameProperty];

[newPerson setValue:@"Doe"
 forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];
[newPerson release];

[addressBook save];
```

Sample Code: ABAddressBook

```
...
[newPerson setValue:@"John"
  forProperty:kABFirstNameProperty];

[newPerson setValue:@"Doe"
 forProperty:kABLastNameProperty];

[addressBook addRecord:newPerson];
[newPerson release];

[addressBook save];
```

ABAddressBook

- Searching

```
-(NSArray*)recordsMatchingSearchElement:
                                (ABSearchElement *)search;
```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

ABSearchElement

- Building blocks for queries
- Example:
  - Find all people who live in New York and have a mac.com email address
  - Need three search elements
    - S1: “Home City = New York”
    - S2: “Email contains mac.com”
    - S3: S1 and S2

ABSearchElement

- Class method on ABPerson and ABGroup

```
+(ABSearchElement*)
    searchElementForProperty:(NSString*)property
                       label:(NSString*)label
                         key:(NSString*)key
                       value:(id)value
                  comparison:(ABSearchComparison)comparison;
```

Sample Code: ABSearchElement

- Search for all people that have “John”
  as first name

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
ABAddressBook *addressBook;
ABSearchElement *element;
NSArray *result;

element = [ABPerson
    searchElementForProperty:kABFirstNameProperty
                       label:nil
                         key:nil
                       value:@”John”
                  comparison:kABEqualCaseInsensitive];
...
```

Sample Code: ABSearchElement

```
...

addressBook = [ABAddressBook sharedAddressBook];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
...

addressBook = [ABAddressBook sharedAddressBook];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

- Search for all people living in “New York”

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

Sample Code: ABSearchElement

```
…

element = [ABPerson
  searchElementForProperty:kABAddressProperty
                     label:kABHomeLabel
                       key:kABAddressCityKey
                     value:@”New York”
                comparison:kABEqualCaseInsensitive];

result = [addressBook
            recordsMatchingSearchElement:element];
```

ABSearchElement

- More complex search

```
+ (ABSearchElement*)
  searchElementForConjunction:(ABSearchConjunction)conj
                     children:(NSArray *)children;
```

- Conjunction

```

```

API Overview

- ABRecord
- Properties
- ABPerson
- ABGroup
- ABAddressBook
- ABSearchElement
- C and AppleScript APIs

C Based APIs

- Functionality identical to Obj-C
- Uses CF naming convention and style
- Classes are represented by opaque types
  - ABPersonRef
  - ABGroupRef
  - ABMultiValueRef
  - …

C Based APIs

```
ABAddressBookRef addressBook = ABGetSharedAddressBook();


ABPersonRef person = ABPersonCreate();


ABRecordSetValue(person, CFSTR(”John”),
      kABFirstNameProperty);

ABRecordSetValue(person, CFSTR(”Doe”),
      kABLastNameProperty);


ABAddRecord(addressBook, person);
```

C Based APIs

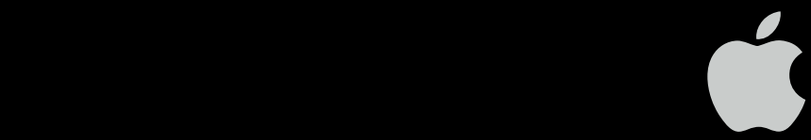
```
ABAddressBookRef addressBook = ABGetSharedAddressBook();


ABPersonRef person = ABPersonCreate();


ABRecordSetValue(person, CFSTR(”John”),
        kABFirstNameProperty);

ABRecordSetValue(person, CFSTR(”Doe”),
        kABLastNameProperty);


ABAddRecord(addressBook, person);
```

C Based APIs

```
ABAddressBookRef addressBook = ABGetSharedAddressBook();


ABPersonRef person = ABPersonCreate();


ABRecordSetValue(person, CFSTR(”John”),
        kABFirstNameProperty);

ABRecordSetValue(person, CFSTR(”Doe”),
        kABLastNameProperty);


ABAddRecord(addressBook, person);
```

C Based APIs

```
ABAddressBookRef addressBook = ABGetSharedAddressBook();


ABPersonRef person = ABPersonCreate();


ABRecordSetValue(person, CFSTR(”John”),
      kABFirstNameProperty);

ABRecordSetValue(person, CFSTR(”Doe”),
      kABLastNameProperty);


ABAddRecord(addressBook, person);
```



C Based APIs

```
ABAddressBookRef addressBook = ABGetSharedAddressBook();


ABPersonRef person = ABPersonCreate();


ABRecordSetValue(person, CFSTR(”John”),
        kABFirstNameProperty);

ABRecordSetValue(person, CFSTR(”Doe”),
        kABLastNameProperty);


ABAddRecord(addressBook, person);
```

AppleScript Support

- Full access to all AddressBook.framework APIs
- Access through the Address Book application

```
tell application “Address Book”
…
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

Sample Script

- List all emails

```
tell application "Address Book"
  set all_people to people
  repeat with a_person in all_people
    set all_emails to emails of a_person
    repeat with a_email in all_emails
      log (value of a_email) as string
    end repeat
  end repeat
end tell
```

What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - Enhanced AppleScript Support
  - People Picker
  - Action Plug-ins

Panther Goals

- API/Property holes
- Performance and bugs
- Developer feedback

What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - Enhanced AppleScript Support
  - People Picker
  - Action Plug-Ins

New ABPerson Properties

- Other Dates property

```
kABOtherDatesProperty (multi-dates)
  kABAnniversaryLabel
```

New ABPerson Properties

- Related names property

```
kABRelatedNamesProperty (multi-strings)
  kABFatherLabel
  kABMotherLabel
  kABFriendLabel
  ....
```

New ABPerson Properties

- Department Property

```
kABDepartmentProperty (string)
```

New ABPerson Properties

- Flags Property
  - Person vs. Company
  - Name ordering (first last vs. last first)

```
kABPersonFlags (int)
```

Notifications Changes

- kABDatabaseChangedNotification and kABDatabaseChangedExternallyNotification now report what changed

```
kABInsertedRecords
kABUpdatedRecords
kABDeletedRecords
```

- Value is an array of UniqueIds

New APIs for ABAddressBook

- Unique Id to Class name

```
- (NSString *)recordClassFromUniqueId:(NSString *)uid;
```

- Returns “ABPerson” or “ABGroup”

New APIs for ABAddressBook

- Default Country Code

```
- (NSString *)defaultCountryCode;
```

- Returns ISO Country code (us, fr, uk, etc.)

New APIs for ABAddressBook

- Default name ordering

```
- (int)defaultNameOrdering;
```

- Returns kABFirstNameFirst or kABLastNameFirst

New APIs for ABAddressBook

- Street Address Formatting
-

```
```

- Attributes
  - Key: address dictionary key (e.g., kABAddressZipKey)
  - Value: Localized description of the key (e.g., “Code Postale” for a French address)

What You'll Learn

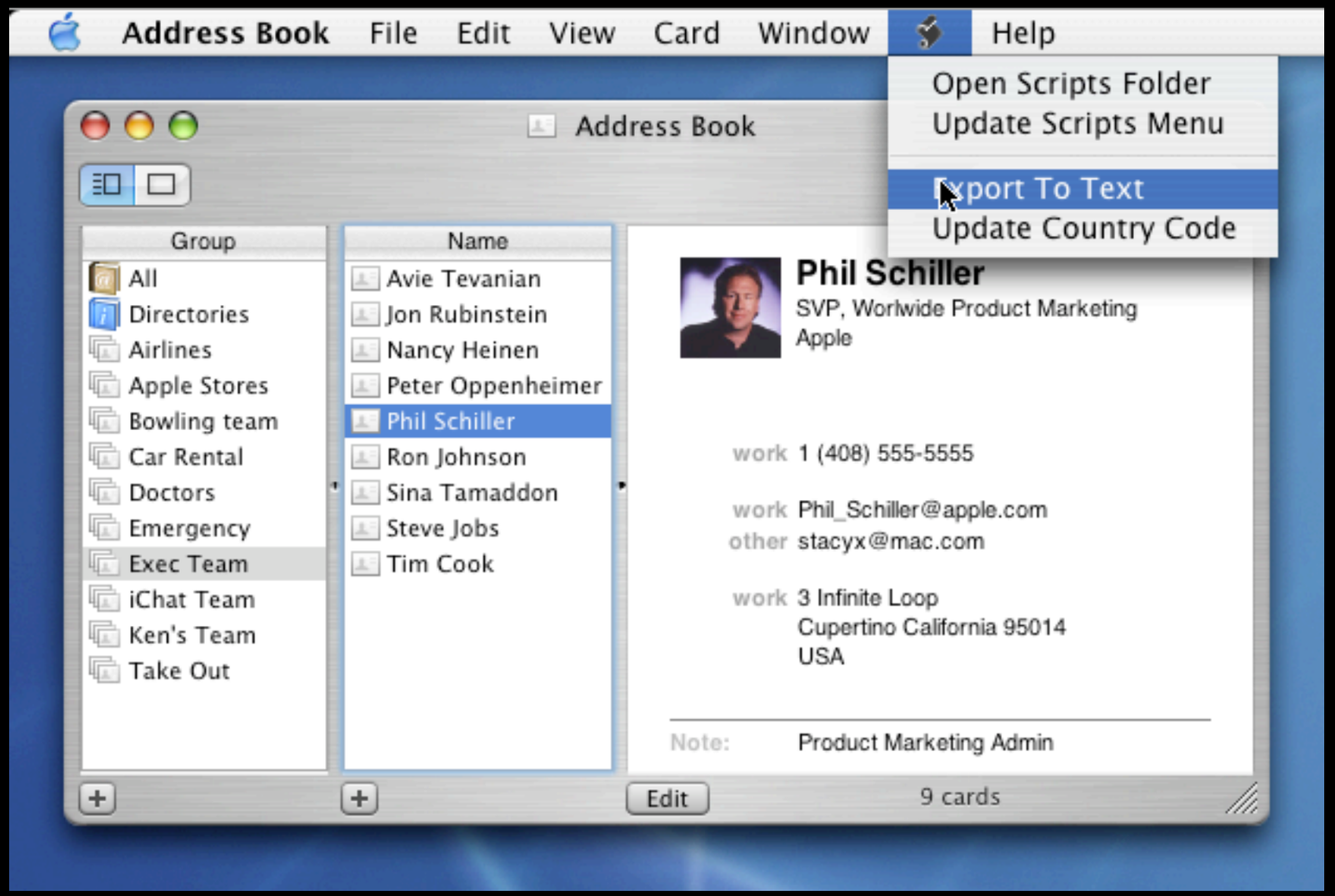
- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - Enhanced AppleScript Support
  - People Picker
  - Action Plug-ins

Enhanced AppleScript Support

- Better parity with Obj-C and C APIs
- Support for all new properties
- Can now get and set
  - Me Card
  - Company flag
  - Image
  - Address Book Selection
- vCard exporting

AppleScript Menu

- Scripts located in: ~/Library/Address Book Scripts



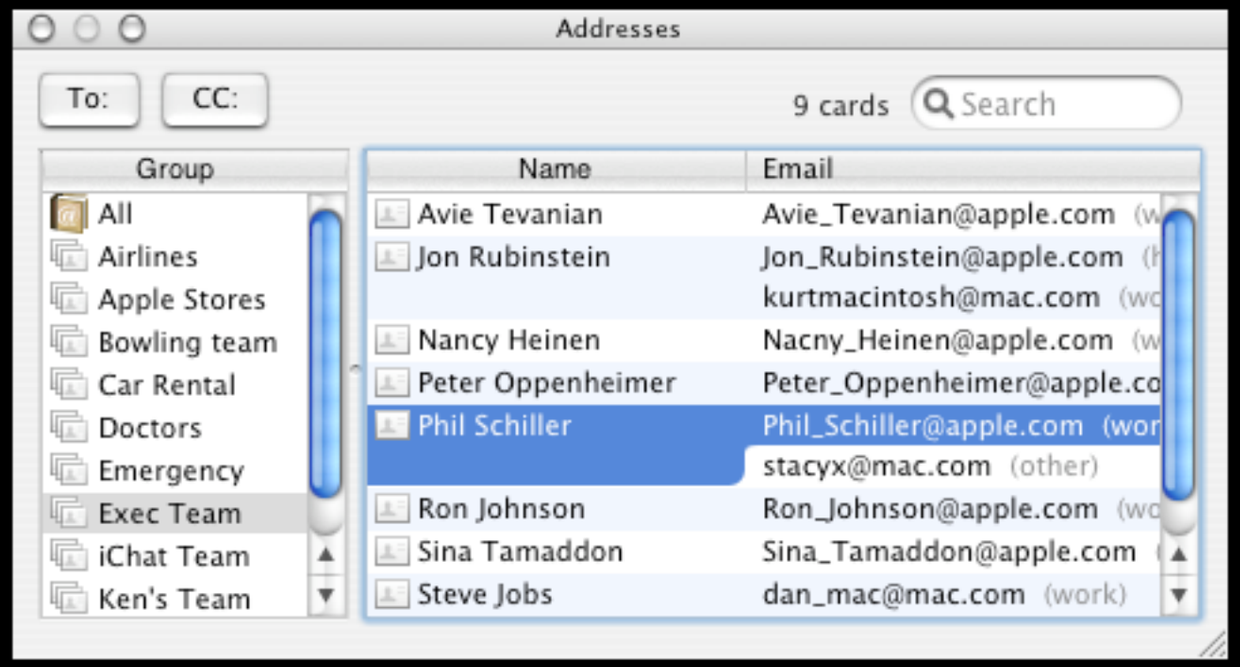
What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - AppleScript Enhancement
  - People Picker
  - Action Plug-Ins

People Picker

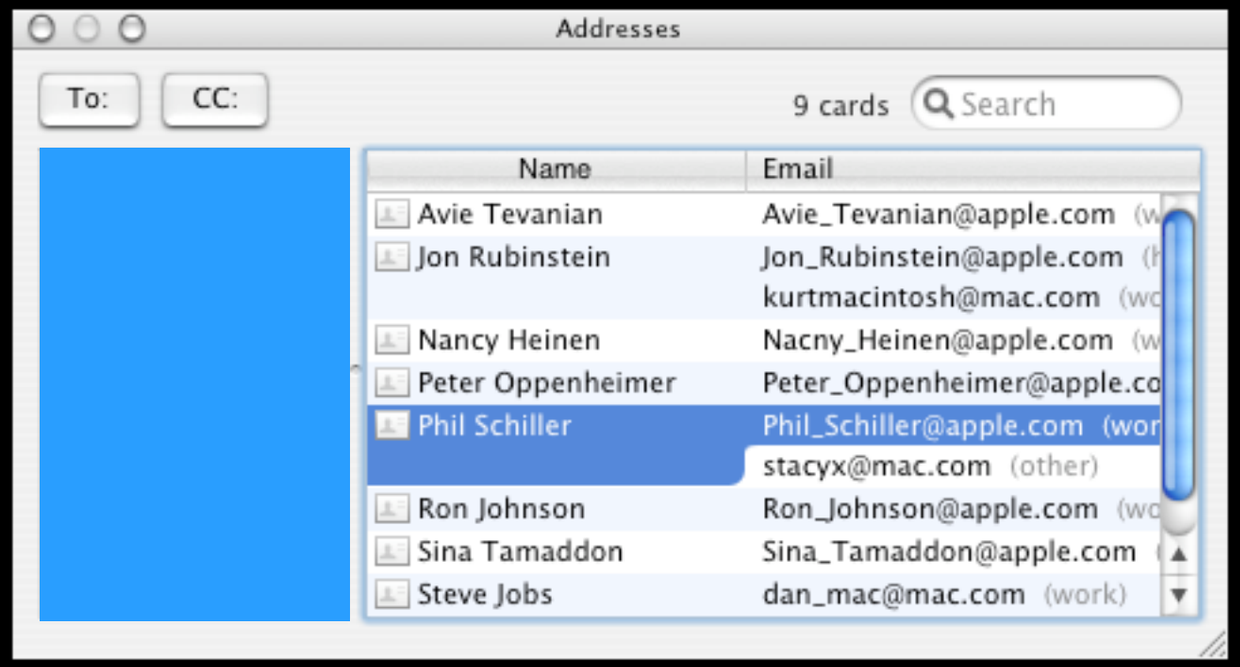
- Provides consistent UI for choosing a person or a group
- In Jaguar used by
  - Mail
  - iChat
  - iCal
- Now public in Panther

People Picker



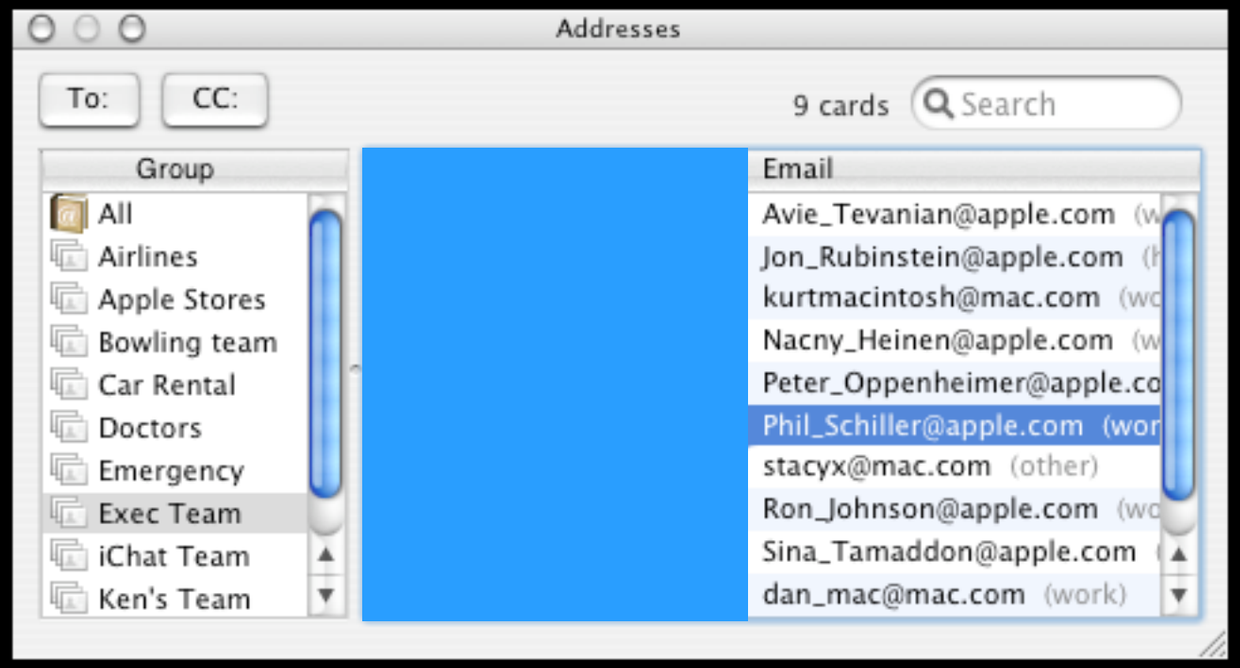
People Picker

- Group column



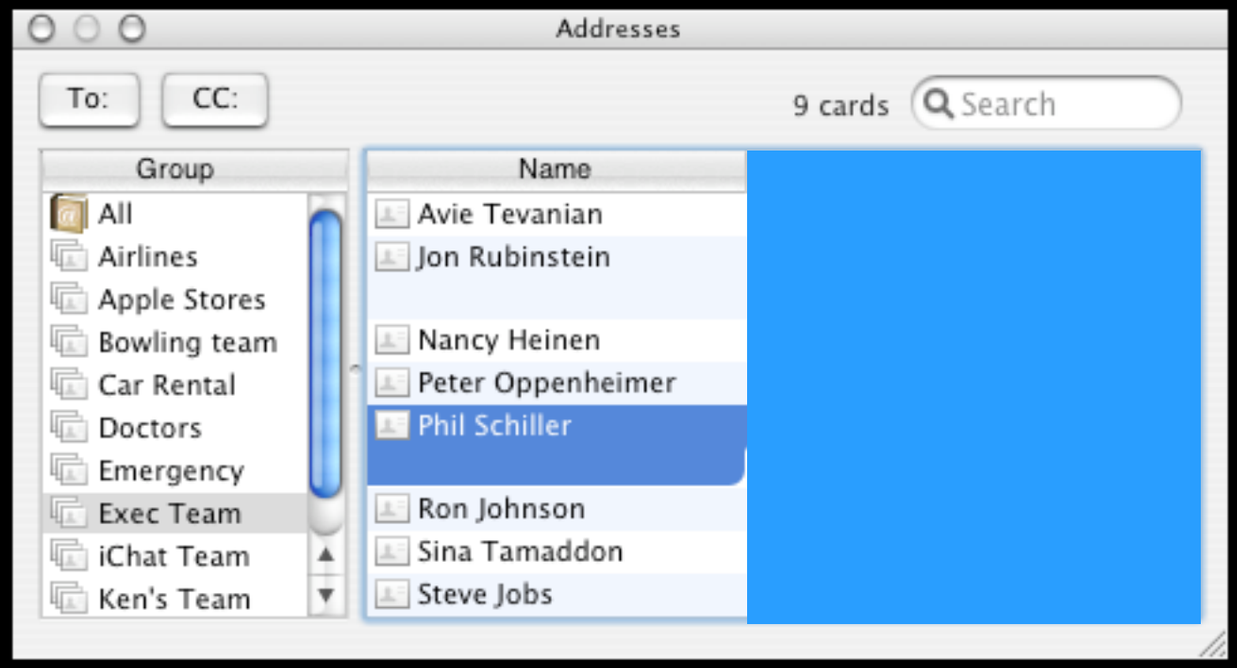
People Picker

- Name column



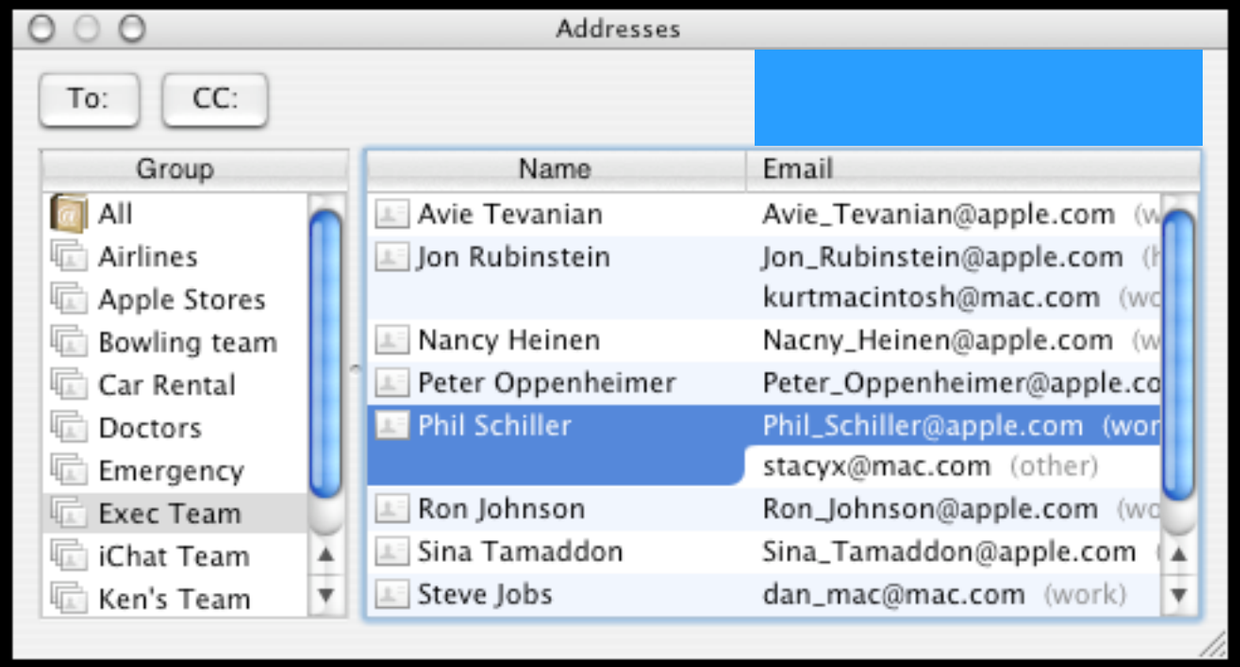
People Picker

- Value column



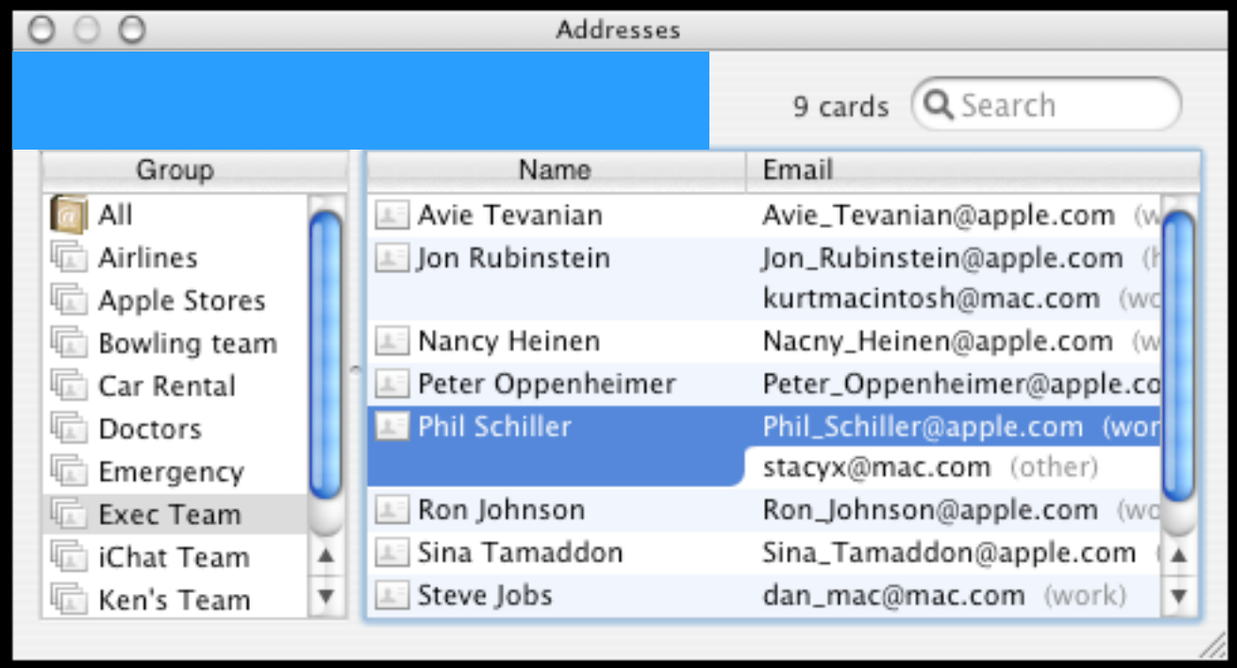
People Picker

- Search field



People Picker

- Auxiliary view (Cocoa Only)



People Picker

- A window for Carbon
  - Header: AddressBook/ABPeoplePickerC.h
- A view for Cocoa
  - Header: AddressBook/ABPeoplePickerView.h
  - Interface Builder Palette available in /Developer/Extras/Palettes

Demo

People Picker Palette

Henri Lamiraux
lamiraux@apple.com

People Picker APIs

- Behavior
- Value column content
- Selections
- Actions

Behavior

- Allow group selection or not
- Allow multiple selection of groups or people
- Value selection behavior
  - No value
  - Single value
  - Multiple values

Value Column Content

- Set and get displayed properties
- Set and get default displayed property

Selections

- Get and set selection of
  - Groups
  - People
  - Values

Actions

- Cocoa
  - Target/Action paradigm
  - Notifications
- Carbon
  - Carbon Events

What You'll Learn

- Address Book Framework Overview
- New in Panther
  - New Properties and APIs
  - AppleScript Enhancement
  - People Picker
  - Action Plug-Ins

Action Plug-Ins

Scott Herz
Address Book Engineering
herz@apple.com

Demo

Action Plug-Ins

Scott Herz

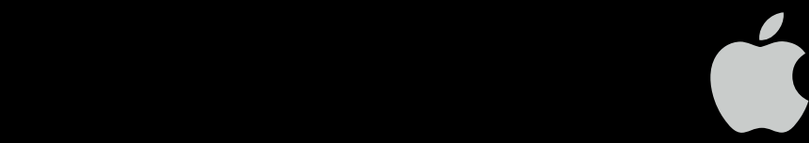
herz@apple.com

Action Plug-Ins

- Actions added to roll-over menu of a given property
- Developer opportunity to extend Address Book

Action Plug-Ins

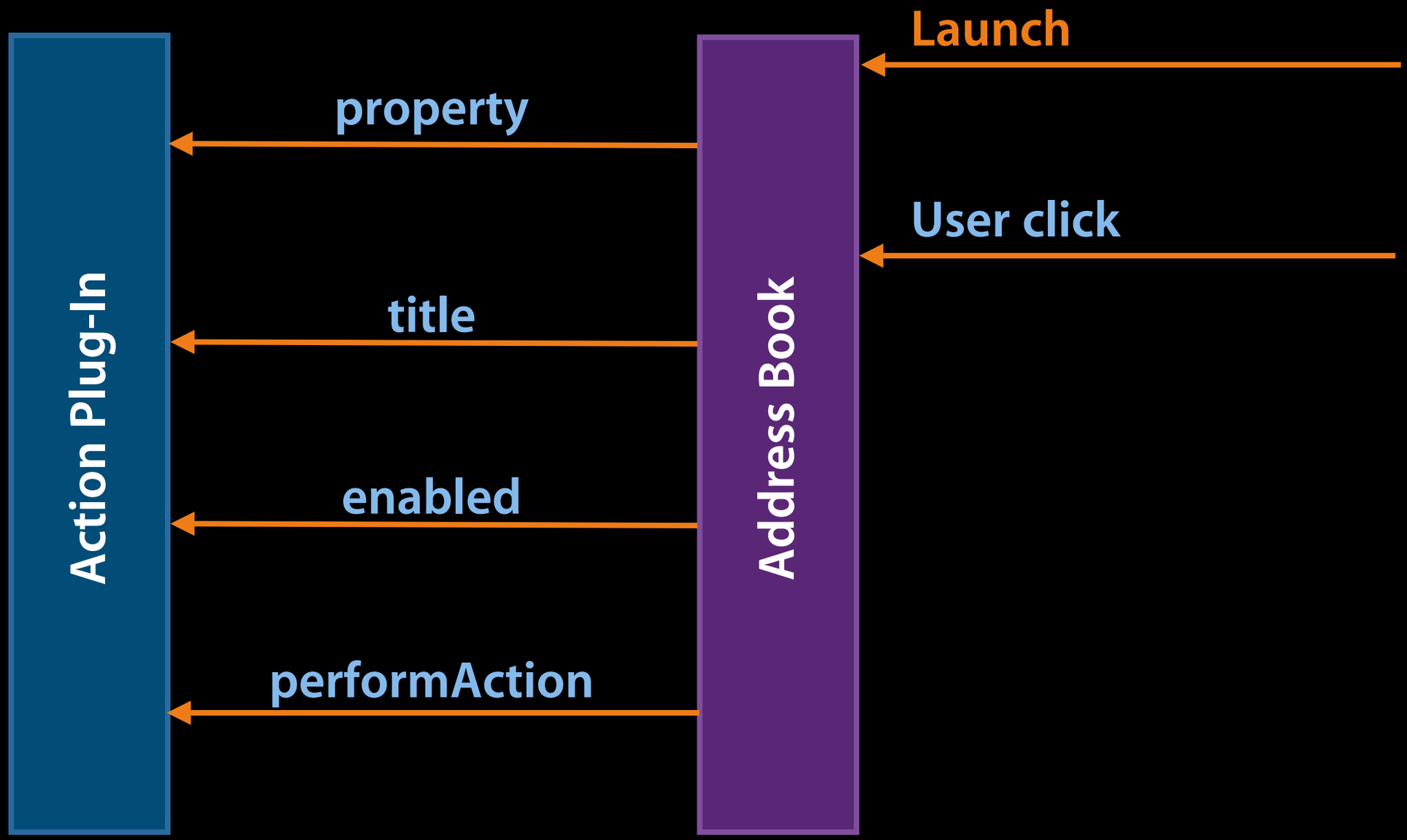
- Can be written in
  - Objective-C
  - C/C++
  - AppleScript
- Headers
  - AddressBook/ABActions.h for Objective-C
  - AddressBook/ABActionsC.h for C/C++



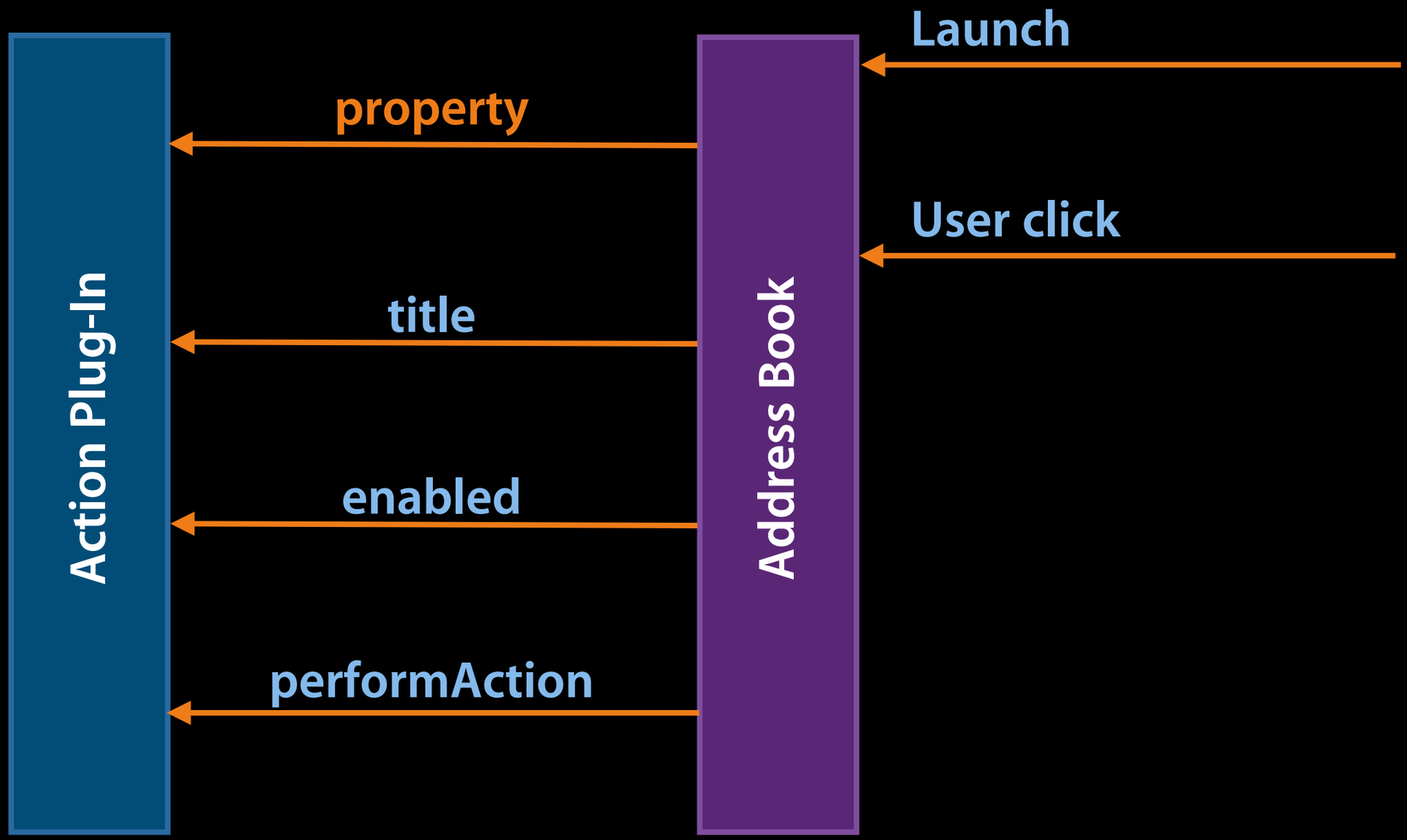
Action Plug-Ins Location

- Plug-ins stored in
  - /Library/Address Book Plug-Ins
  - ~/Library/Address Book Plug-Ins

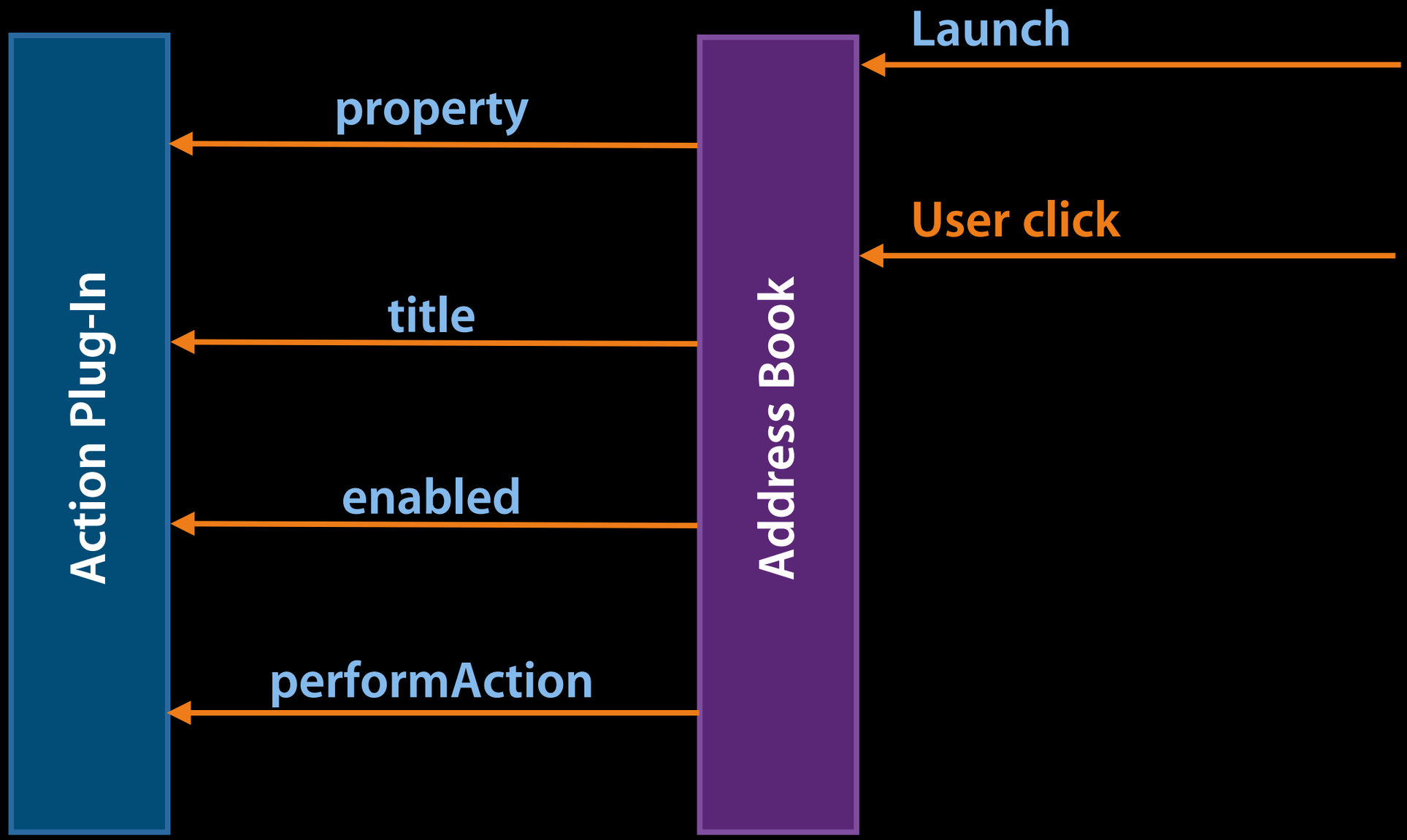
Action Plug-Ins Process



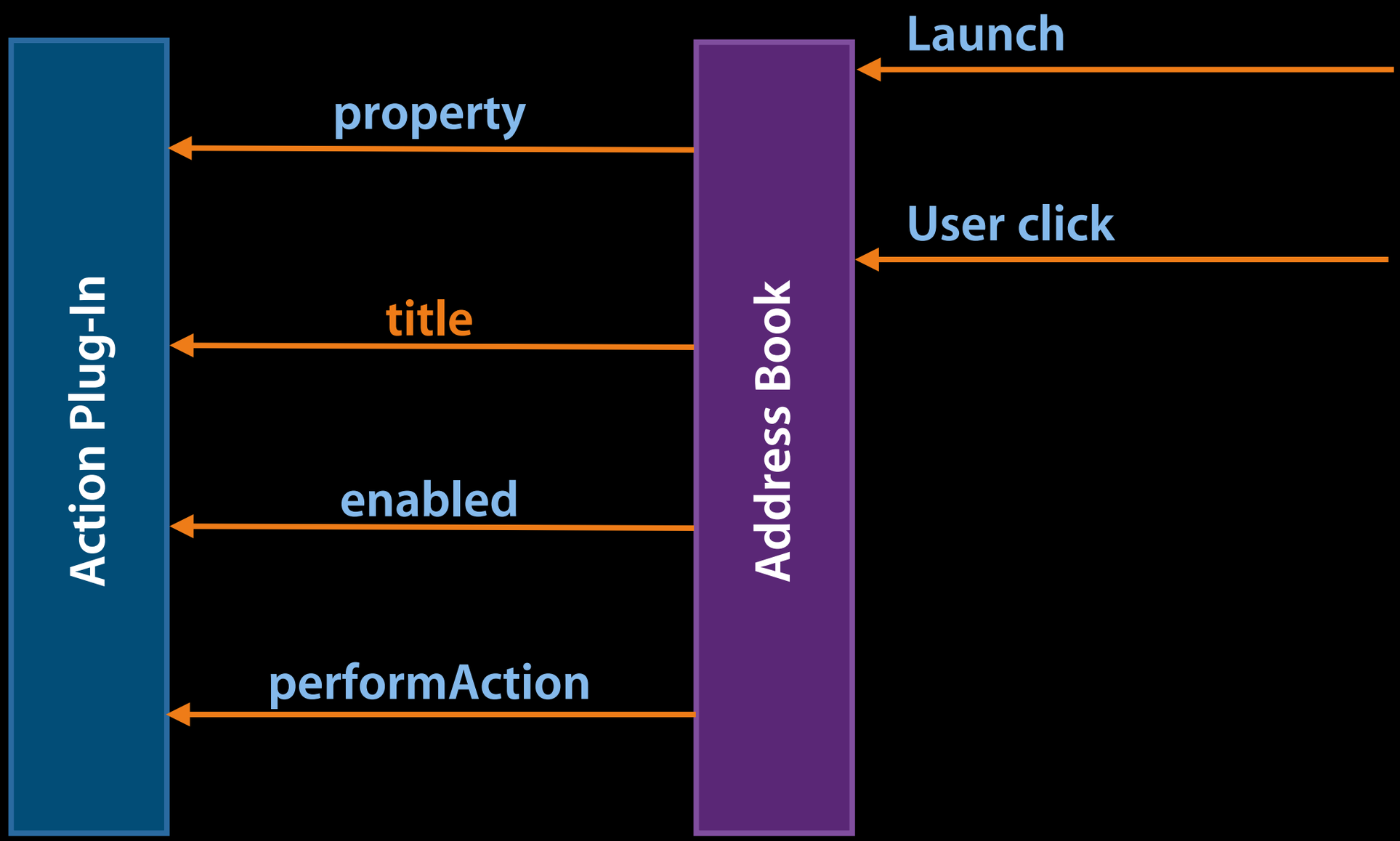
Action Plug-Ins Process



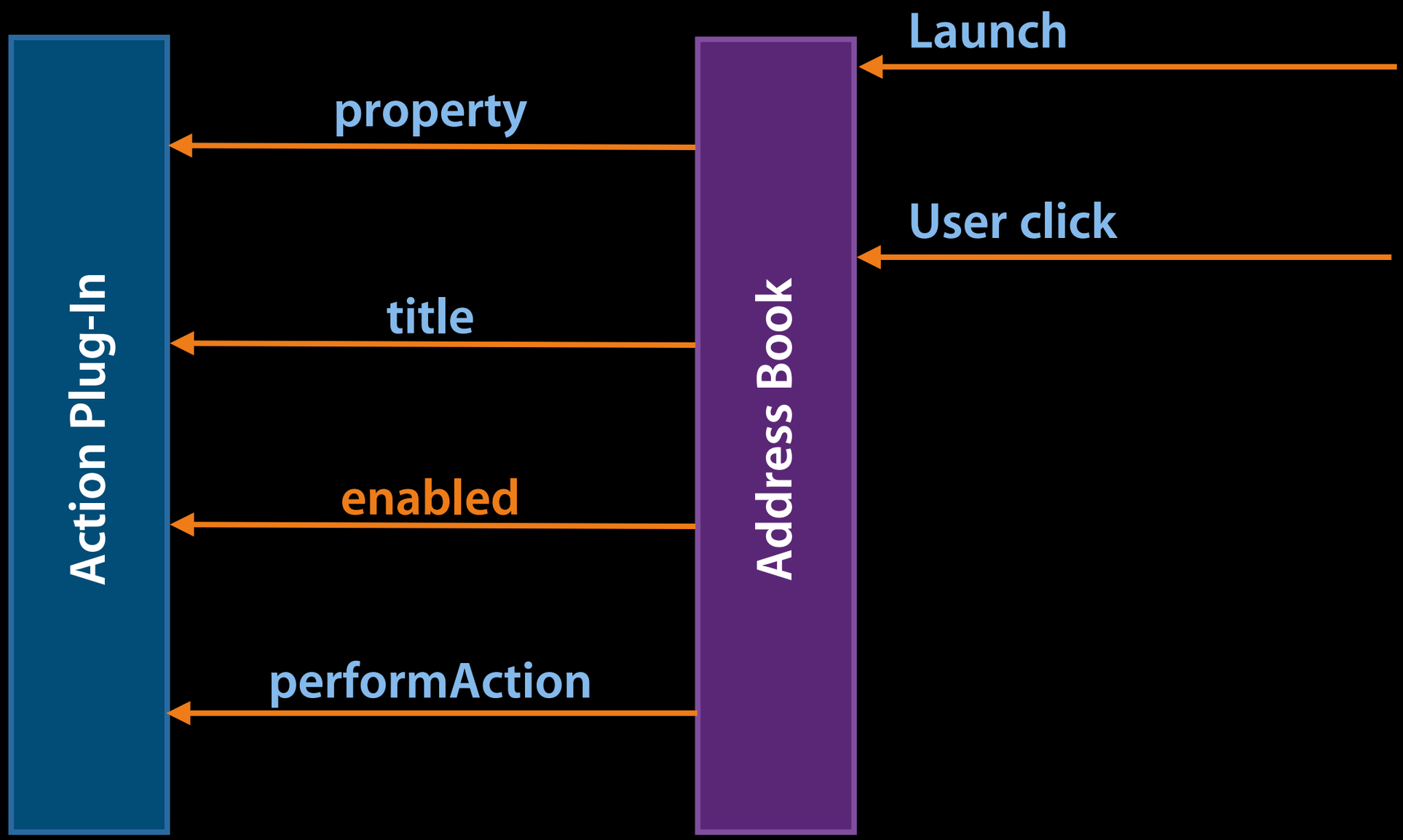
Action Plug-Ins Process



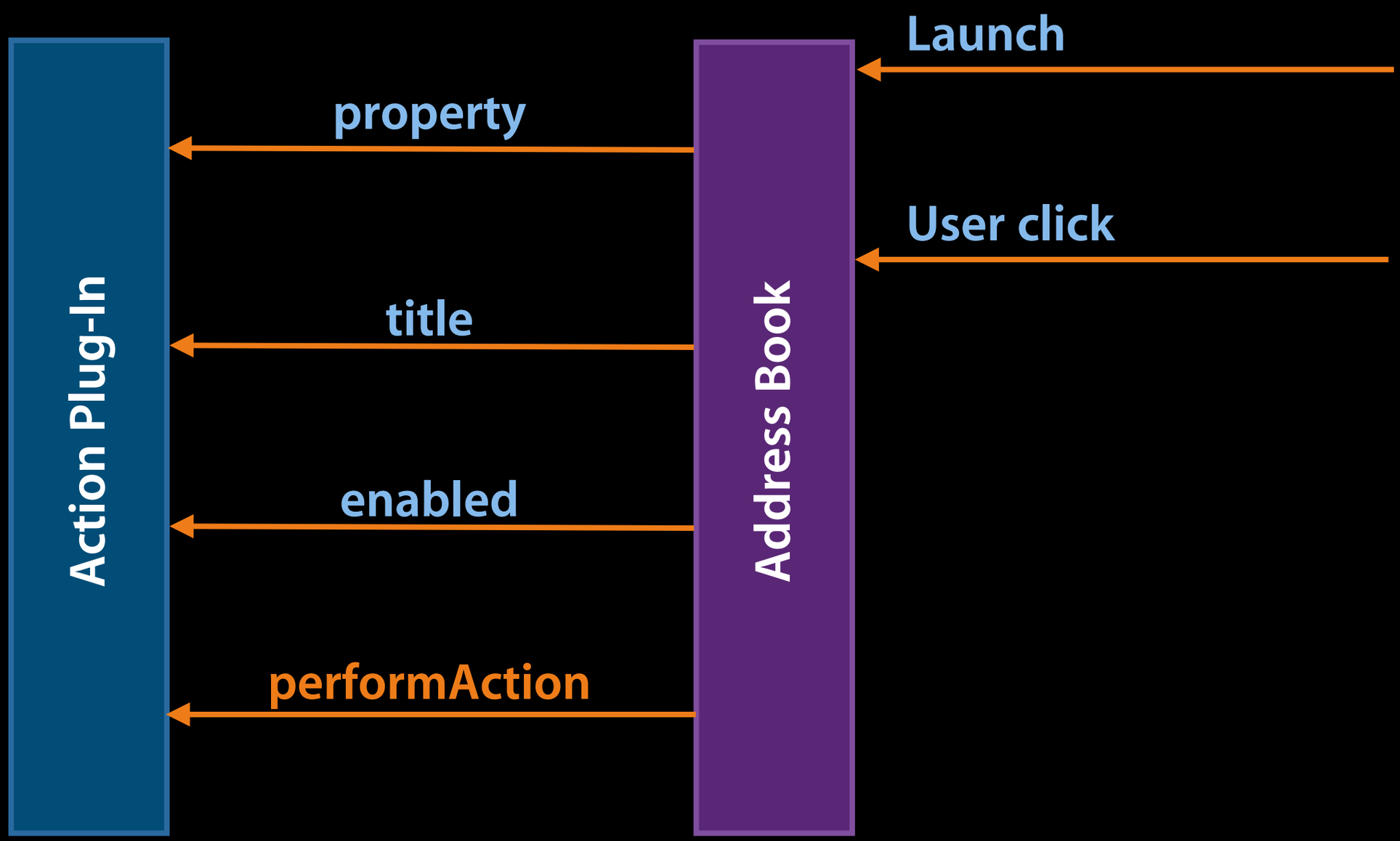
Action Plug-Ins Process



Action Plug-Ins Process



Action Plug-Ins Process



Action Plug-Ins: Objective-C

```
@interface NSObject (ABActionDelegate)

- (NSString *)actionProperty;

- (NSString *)titleForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

- (BOOL)shouldEnableActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier; // Optional

- (void)performActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

@end
```

Action Plug-Ins: Objective-C

```
@interface NSObject (ABActionDelegate)

- (NSString *)actionProperty;

- (NSString *)titleForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

- (BOOL)shouldEnableActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier; // Optional

- (void)performActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

@end
```

Action Plug-Ins: Objective-C

```
@interface NSObject (ABActionDelegate)

- (NSString *)actionProperty;

- (NSString *)titleForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

- (BOOL)shouldEnableActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier; // Optional

- (void)performActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

@end
```

Action Plug-Ins: Objective-C

```
@interface NSObject (ABActionDelegate)

- (NSString *)actionProperty;

- (NSString *)titleForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

- (BOOL)shouldEnableActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier; // Optional

- (void)performActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

@end
```

Action Plug-Ins: Objective-C

```
@interface NSObject (ABActionDelegate)

- (NSString *)actionProperty;

- (NSString *)titleForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

- (BOOL)shouldEnableActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier; // Optional

- (void)performActionForPerson:(ABPerson *)person
identifier:(NSString *)identifier;

@end
```

Action Plug-Ins: C

```
ABActionCallbacks* ABActionRegisterCallbacks(void);


typedef struct {
    CFIndex                         version;
    ABActionGetPropertyCallback     property;
    ABActionCopyTitleCallback       title;
    ABActionEnabledCallback         enabled;
    ABActionSelectedCallback        selected;
} ABActionCallbacks;
```

Action Plug-Ins: C

```
typedef CFStringRef (*ABActionGetPropertyCallback) (void);

typedef CFStringRef (*ABActionCopyTitleCallback) (ABPersonRef
  person, CFStringRef identifier);

typedef Boolean (*ABActionEnabledCallback) (ABPersonRef person,
  CFStringRef identifier);

typedef void (*ABActionSelectedCallback) (ABPersonRef person,
  CFStringRef identifier);
```

Action Plug-Ins: AppleScript

```
using terms from application "Address Book"

    on action property
        ...
    end action property

    on action title for aPerson with aValue
        ...
    end action title

    on should enable action for aPerson with aValue
        ...
    end should enable action

    on perform action for aPerson with aValue
        ...
    end perform action

end using terms from
```

Conclusion

John Geleynse
Manager, Software Evangelism and User Experience Evangelist
geleynse@apple.com

Conclusion

- Widely used in Jaguar
- Trend will continue in Panther
- People Picker for easy access to contacts
- Action Plug-Ins and Script menu to extend the Address Book application

For More Information

Reference Library

Documentation

Address Book Framework

http://developer.apple.com/ue/addressbook/

Wrap Up

Roadmap

400 Introduction to Developing Applications with Cocoa	Nob Hill Mon., 5:00pm
000 Mac OS X State of the Union	The Presidio Mon., 2:00pm
001 Deliver a Complete Mac OS X User Experience	The Presidio Mon., 5:00pm
423 How to Write a Modern Carbon Application	Nob Hill Fri., 2:00pm

Who to Contact

John Geleynse
Manager, Software Evangelism
geleynse@apple.com

http://developer.apple.com/wwdc2003/urls.html

Q&A

Address Book Engineering Team

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference